Macromedia Central Internet Applications—United and Unplugged

Session 731

Macromedia Central Internet Applications—United and Unplugged

Ursula Ringham
Partnership Manager
Worldwide Developer Relations

Macromedia Central Internet Applications—United and Unplugged

Dave Calaprice,
dave@macromedia.com

Michael Williams
mwilliams@macromedia.com

Alex Wright
alex@agwright.com

Macromedia Central Internet Applications—United and Unplugged

Dave Calaprice,
dave@macromedia.com

Introduction

- Today's presentation will provide an overview of Macromedia Central, a new environment for internet-enabled applications
- We'll talk about how Macromedia views the rich Internet applications space and solutions that we are working on

What You'll Learn

- We'll talk about key challenges in the market today, and the ways that Macromedia Central will address these issues
  - Situation overview
  - Macromedia Central description and framework
  - Process for developing and deploying applications in Macromedia Central

Technology Framework

- This presentation addresses the following technology areas:
  - Rich internet applications
  - Web services and standards
    - XML, SOAP, WSDL
  - Macromedia Flash

The Macromedia Vision

- Great experiences build great businesses
- Business users can create great experiences
- People want those great experiences wherever they are

Rich Internet Applications

Best of
the Desktop

Rich Internet
Applications

Best of
the Web

Best of
Communication

What's Issues Does Central Address?

- Internet of today is document centric
- Installation of applications from Internet is clumsy and cumbersome
- No context for applications to work together
- No data availability for offline use
- No consistent way of monitoring events/notifications

Macromedia Central Overview

Macromedia Central is an efficient internet-enabled platform, which you can use even when you're not online

- **Central** environment for collecting and managing apps
- **United** applications work together
- **Unplugged** stay productive even when you don't have a web connection

Customer Benefits of Central

- Combines the best of:
  - Desktop interactivity and availability
  - Web deployment and up-to-date information
- Cooperative applications work together and share data
  - Delivers efficient experience for customer
- Fast, easy installation of Central and applications
- Built-in try/buy mechanisms for applications
  - Risk free environment encourages exploration
- Use applications online and offline
- Central available on OSX, Windows and handheld devices

Developing for Macromedia Central

- Use Macromedia Flash MX and the Central SDK to develop applications
- Use existing components, plus additional Central components
- Build on top of Web services or other structured data
- Central environment built upon Flash player technology

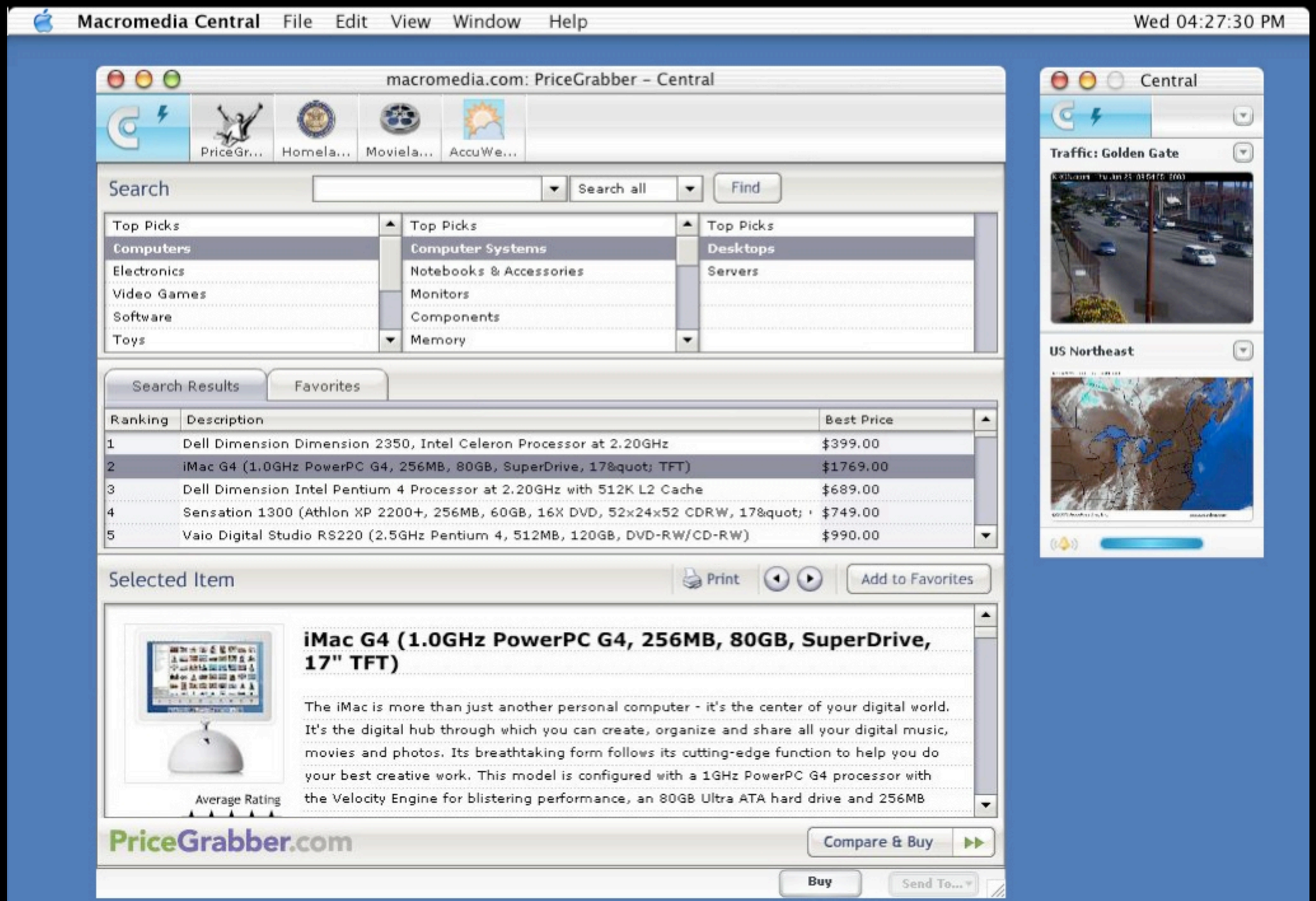
Developer Benefits

- Easy to use, widely supported development tool
- Central provides built-in services and support
  - Transaction infrastructure
  - Download and installation functionality
  - Services for exchanging data
  - Auto-updating technology
- Macromedia provides broad distribution of Central platform and application finder to deliver exposure for Central applications

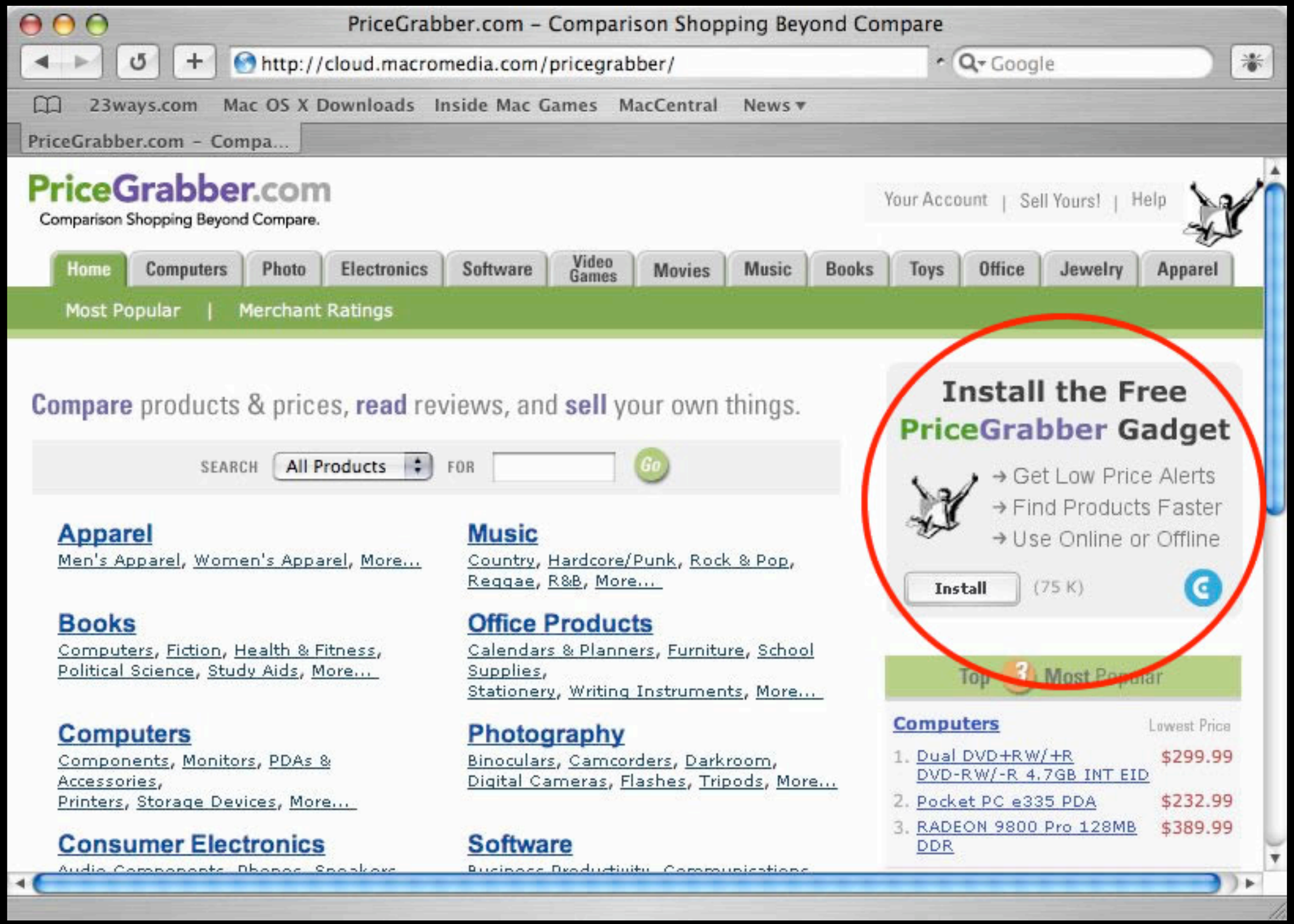
Benefits of Flash Technology

- Flash ubiquity—player distributed to all desktop systems over the past 5 years
- Over 750K Flash developers worldwide today
- Portability across multiple platforms—from desktop to handheld
- Flash player distributed on virtually all PC systems
- Ease of development
- Very space efficient file format and architecture
- Built in support for multiple media types
  - Digital video, audio, animation, images, interactivity and communications server

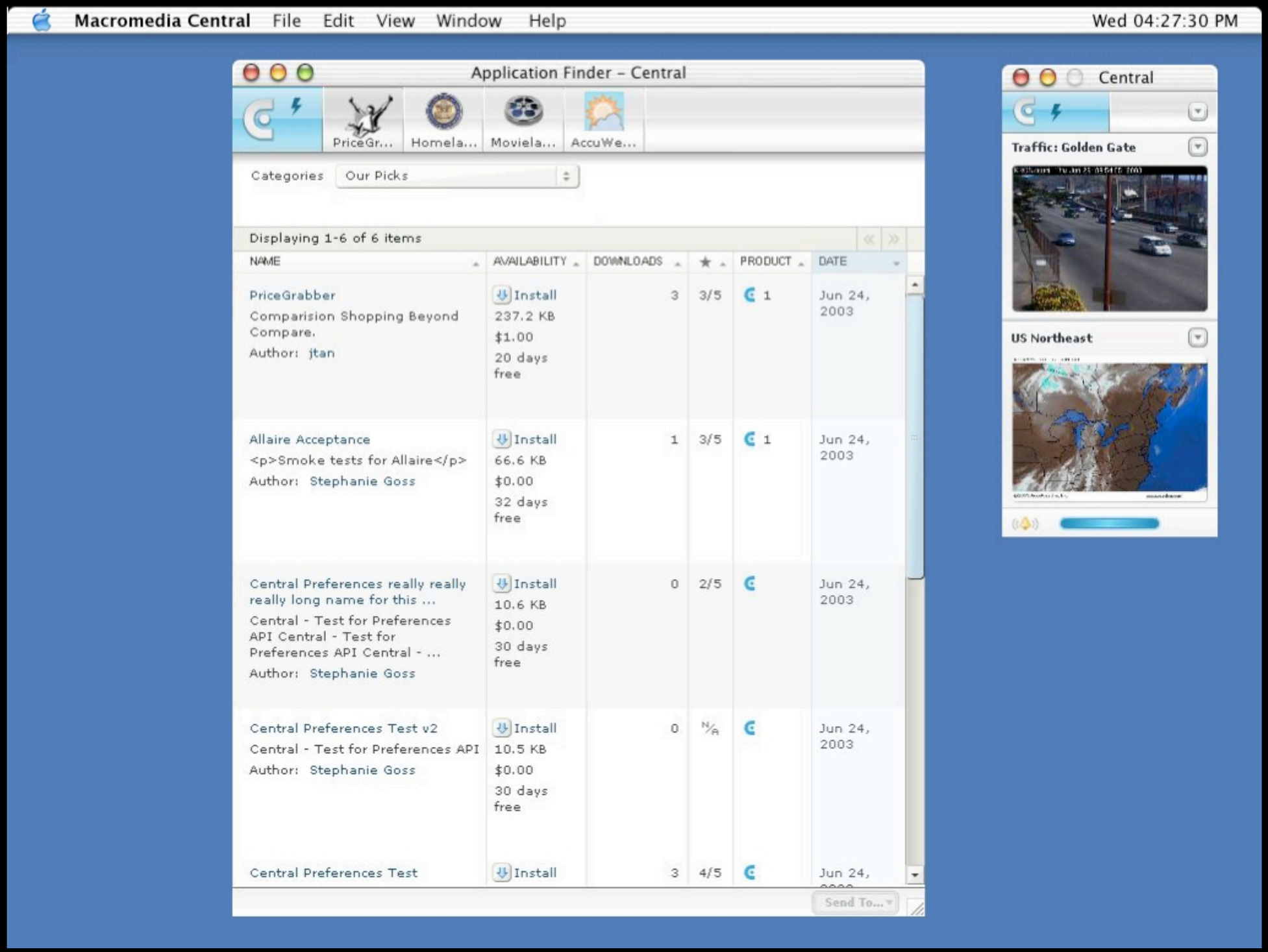
User Interface



Install Instantly From a Web Site



Or Use the Application Finder



Demo

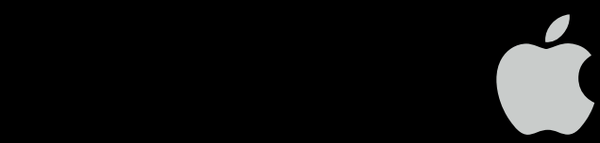
Alex Wright
alex@agwright.com

How It Works

Technical Overview

What Is SWF

- SWF (Small Web Format) is the runtime format of the Flash Player
- Compiled binary format
- Contains custom byte code, media, and vector data
- Open format maintained my Macromedia
- Published by hundreds of multimedia authoring tools and web deployment environments
- Primary scripting language is "ActionScript"
  - Based upon ECMA-262 with extensions for custom DOM



Central Application Architecture

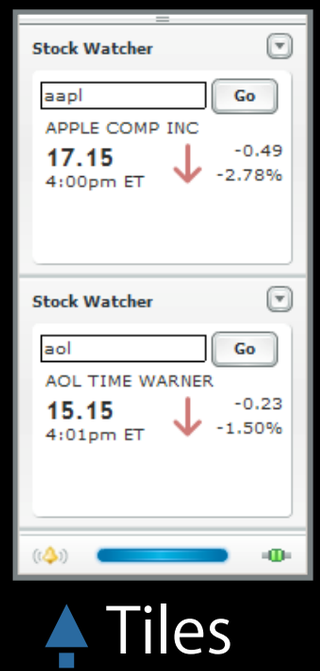
- Application shell
- Console
- Central UI components
- Elements of a Central application
  - Main application
  - Tiles
  - Agents
  - Notices

Flash Player Foundation

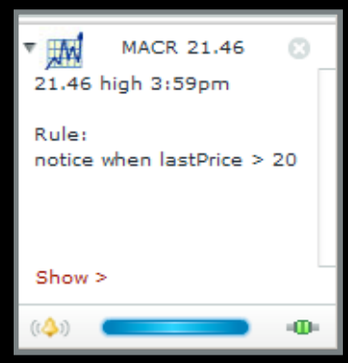
- Built on Macromedia Flash Player 6 code base
  - + Performance enhancements
  - + WebService APIs
  - + Regular expressions APIs
  - + UrlCaching APIs
  - + E-commerce enabled

Central Application Architecture

Application Window



Tiles



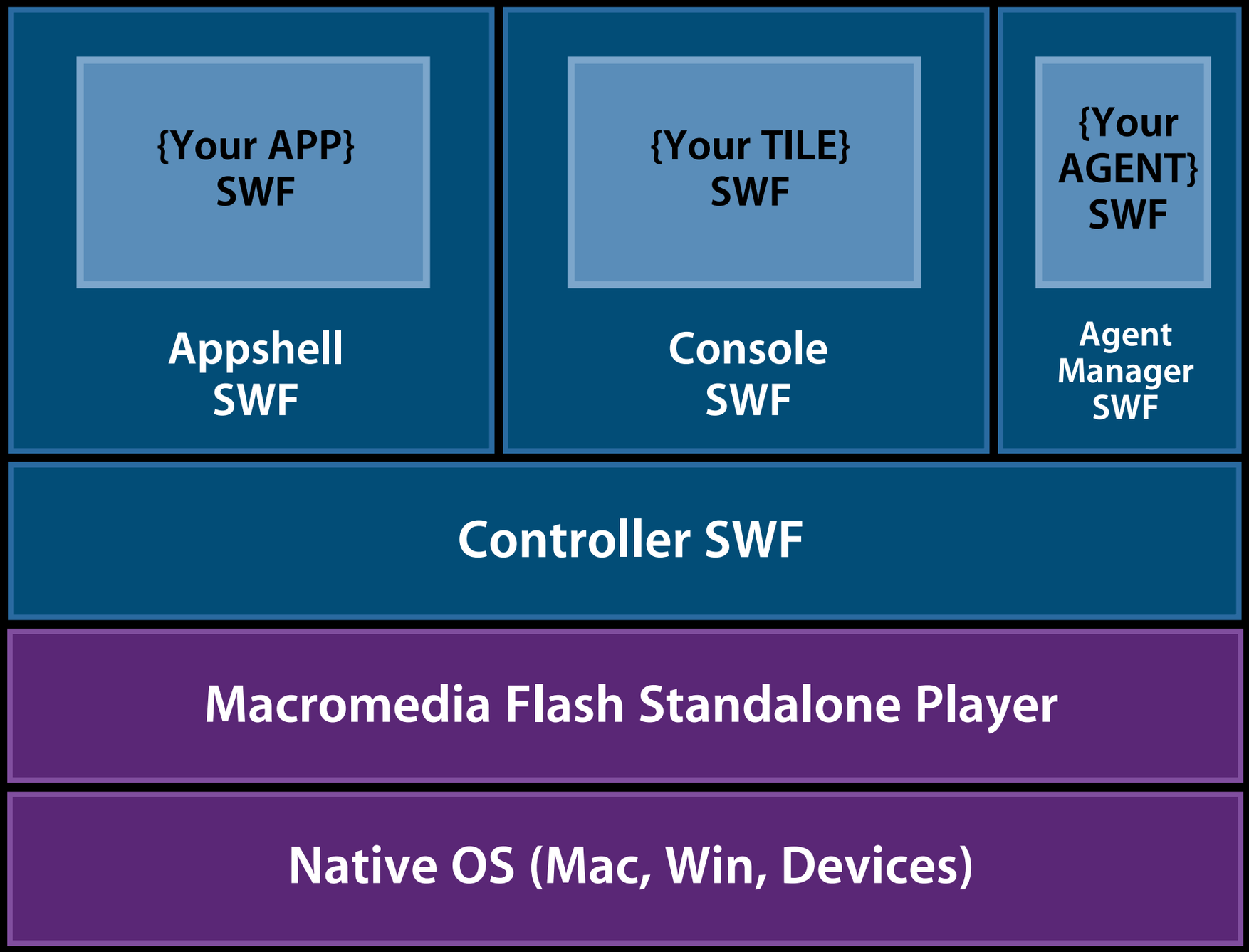
Notices

Agent
(Data Management)

Stocks Web Service

Local Storage

Central Stack



An Application

- Contains three primary parts
  - XML product definition (product.xml)
  - Compiled Application (SWF)
  - Application Icon (SWF, JPG)

Product XML

```
<product name="empty_app"
        created="6/6/2003"
        version="1"
        lifetime=""
        price="Free"
        requireVersion="1"
        help="http://www.mysite.com/empty_app_help.html"
        category="Productivity"
        vendor="www.mysite.com"
        vendorID="myvendorID"
        productID="CND100-86032-27221-59336">
        <author name=”Michael Williams" info="http://www.macromedia.com/"/
>
        <description>Empty Application Shell for Central</description>
        <icon src="empty_app.jpg" size="37"/>
        <application name="Empty App"
                src="empty_app.swf"
                width="550" height="400"
                help=" http://www.mysite.com/empty_app_help.html "
                enabled="true"
                lang="en"
                background="#FFFFFF"
                version="1"
                sendto="true">
                 <icon src="empty_app.jpg" size="37"/>
        </application>
</product>
```

SWF Communication

- All SWF communication takes place over a reserved synchronous connection between the controller and each SWF
- Utilizes “localConnection”
- Allows you to communicate between any SWF even those in a web browser
- Same method used to initiate Central framework and custom application downloads

Controller

- Central hub for app messaging
- Manages new application shells, agents and tiles
- Stores and manages global preferences

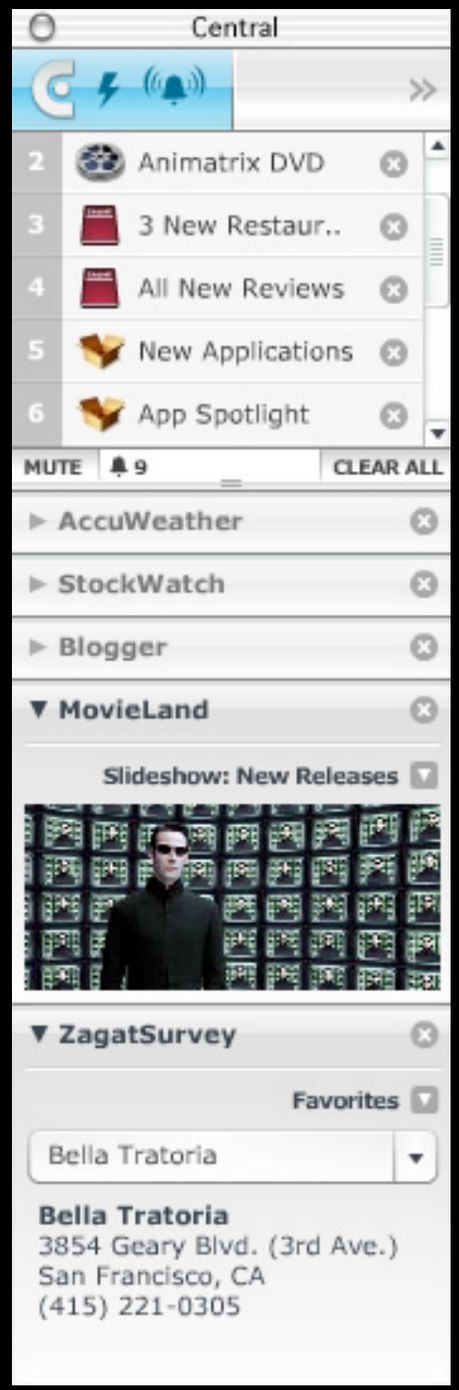
Agents

- SWF files with no visual representation
- Script only used for background processes
- Allows for persistent data access and management without the presence of your application
- Managed by the Controller AgentManager
- Each app can have at most 1 agent

```
<agentClass name="stockAgent" src="app_agent.swf"/>
```

Console

- The console is a persistent area for your application to exist on the users desktop
- Consists of two primary views
  - Tiles
  - Notices

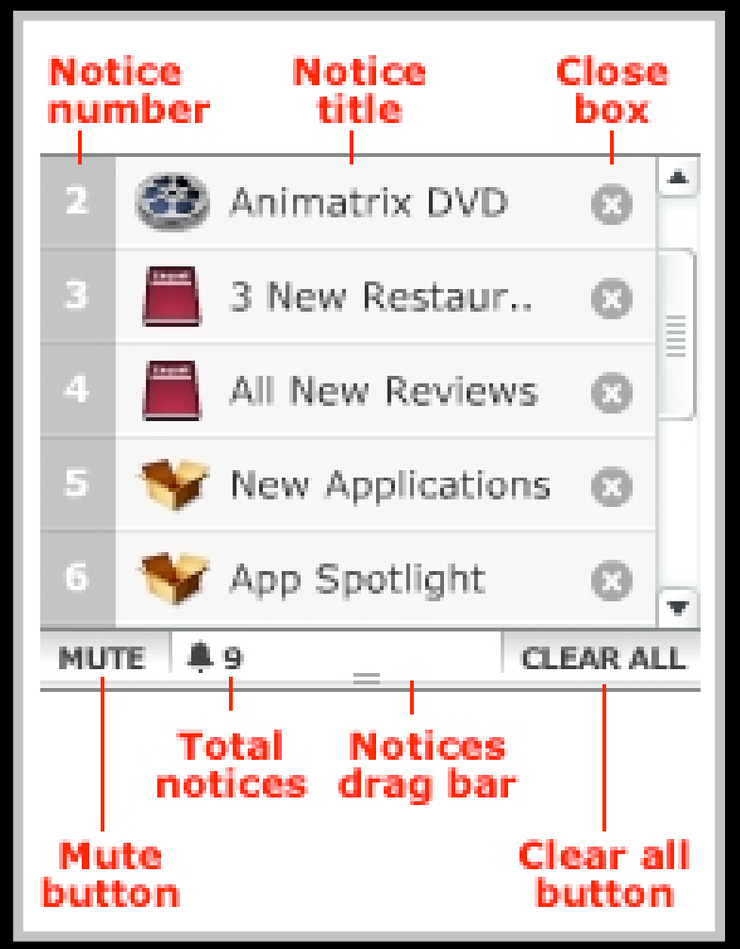


Tiles

- Tiles
  - Framework that provides a constant desktop presence for their application
- Tiles can be updated via agents for up to the minute data even when the application isn't running
  - A weather Tile showing your local weather
  - A Sports Tile showing your sports preferences defined in a custom sports application

Notices

- Notices
  - A standard mechanism to raise visual alerts regarding a change in the state of an application
- Notices are displayed within the console and triggered by events within an agent

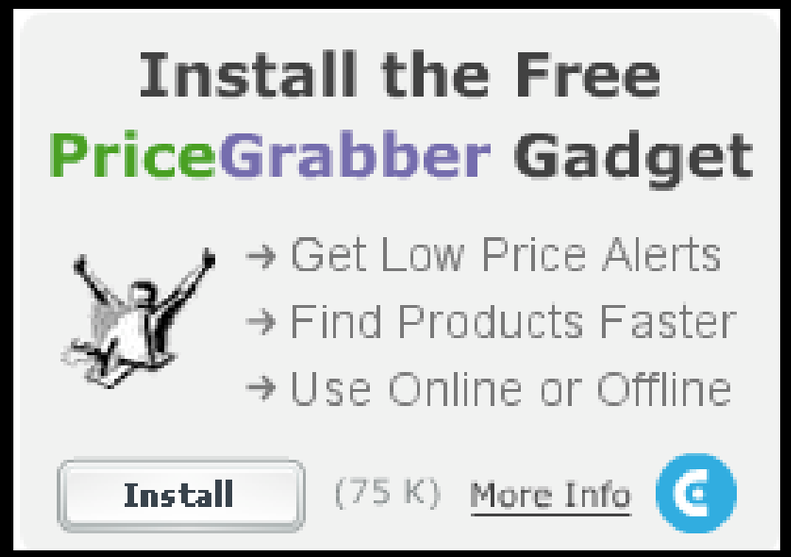


Application Shell

- Primary staging area for 3rd party content
- Central location for application collections and application interaction
- Developer has APIs for
  - Window size, resize events, tab index, status bar, preferences, activation(open), deactivation(close)
  - Network status, local internet cache
  - Notices, agents, and tile creation/destruction

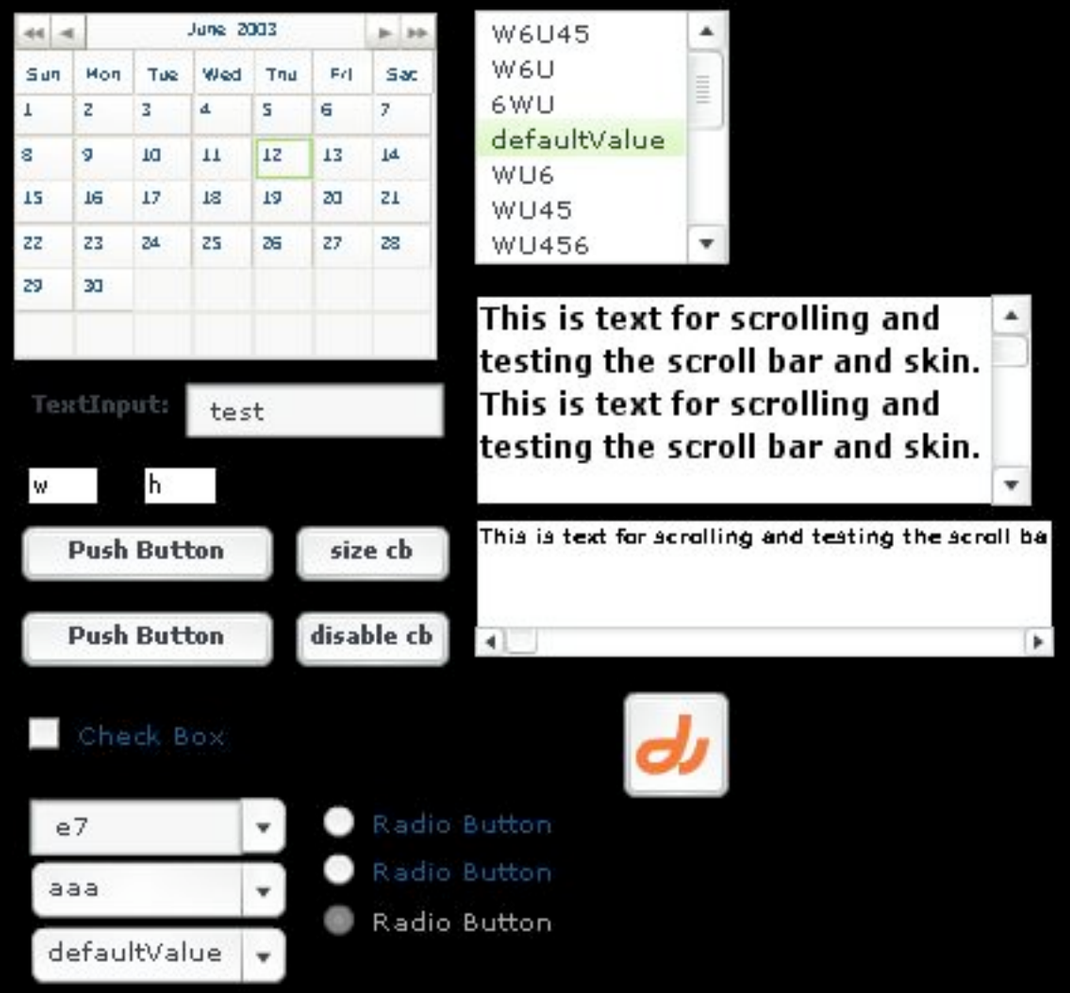
Packaging Your App for Distribution

- SWF file resides at your site
- Product XML descriptor file
- “Installer” files
  - SWF button embedded in HTML (ex: “installer.swf”)
- Submission to Macromedia for listing in public application finder



Central Interface Components

- Central Components
  - Standard UI elements such as Radio button, checkbox, etc.



Halo

- Consistent look and feel across all Macromedia tools and the content they build
- Central components are linked to a Runtime Shared Library (RSL) on macromedia.com
  - RSL URL is cached by Central
  - Your app's download is smaller
  - Your app picks up any bug fixes automatically
- Central uses these same components; Use them to make your app looks like part of Central

Consuming and Storing Data

- XML support built into Macromedia Flash Player 6
- Native web services support
  - No application gateway or Flash Remoting required
  - SOAP and WSDL 1.1
- Local Shared Objects
  - ActionScript structures
  - Text data
- LocalInternetFiles()
  - Any file type (JPG, XML, etc.)

Building for Central

Integrating Your App Into Central

- Basics
  - Give Central your phone number:
    - Add Central.initApplication(this,this) to your app; This tells Central you are here, gives it an object to call notification methods on
  - Central calls you back with its phone number:
    - onActivate = function(shell, appId, shellId, baseTabIndex, appData) is called once Central gets initApplication; Save the Shell callback object to access Central services
- See DevKit for
  - Other notification methods Central can call on your app (if you provide them
  - Shell methods your app can call to access Central functionality

Installation

- Create your product.xml (or modify an existing one to meet your needs)
- Place your app.swf, product.xml, icon.jpg, the installer.swf, and the installer.html files in a directory on your web server. Make sure they are all in the same directory
- Go to page, click on button = Installed!
- Modify installer.html to suit your needs
- Use Central Application Publisher to register commercial applications, they will show up in the Application Finder for distribution

LocalConnection Is Your Friend

- Simple API
  - gCom = new LocalConnection();
  - gCom.connect("StockAgent");
  - gCom.getTicker = this.myGetTicker
  - gCom.send("StockAgent", “getTicker”, arg);
- Use to communicate between
  - Application and Agent
  - Application and Tile
  - Tile and Agent
- Use to debug
  - Set up debug text field in standalone player, looking for input from a Central based App/Tile/Agent’s LocalConnection request (see Tracer SWF in DevKit)

Add a Tile

- Create a new SWF for your tile
- Set up the phone call
  - Central.initTile(this, this);
  - function onActivate(console, tileid, viewerId, position, baseTabIndex, data); Save Console callback to access Central functionality
- Installation: add tile.swf to your product.xml
- See DevKit for
  - Other notification methods Central can call on your tile (if you provide them
  - Console methods your app can call to access Central functionality)
- An application can have multiple tiles

Use Tiles for

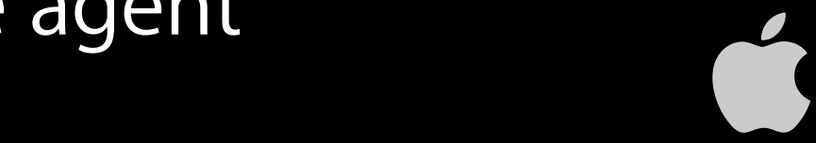
- Live monitoring of pushed data such as current stock price, current sports score
- Quick access to subset of app functionality
  - Sometimes easier to bring up tile than app: look up phone #, quick calculation
  - Sometimes want to use one app while another is in use; Look up movie times while in an invitation planner
  - Really cool when Tiles dynamically show info related to what's going on in the active application; Calendar tile dynamically showing you what else is going on while in an app which lets you select a day to make a reservation on

Don’t Use Tiles For

- Necessary parts of your main apps functionality
  - Awkward to push other tiles out of the way whenever your app activates
  - Awkward to require two separate windows to use app
- Notification of events (we have a feature for that)

Add an Agent

- Create a new SWF for your agent
- Set up the phone call
  - Central.initAgent(this, this);
  - function onActivate(agentManager, agentID, initialData);
    Save agentManager callback to access Central functionality
- Installation: add agent.swf to your product.xml
- See DevKit for
  - Other notification methods Central can call on your agent (if you provide them)
  - agentManager methods your app can call to access Central functionality
- Each application can have at most one agent



Use Agents For

- Persistent monitoring of data you want notification of
- Data management (more later)
- Communication with other apps which know your LocalConnection (more later)

Don’t Use Agents For

- Processor intensive tasks, unless that work is throttled down when your app is not active
  - Badly behaved agents may make users disable agents from the preferences settings
  - If necessary, you can ask Preferences if agents are enabled and put up a dialog indicating that your app will not function correctly with agents turned off

Create a Notice

```
Tell Shell (or AgentManager or Console) callback to add notice:
      notice = new Object();
      notice.name = "New Test Notice!";
      notice.description = "This is the notice message.“

      // set engage link name
      notice.engageString = "Show";

      // set that application should open when engaged
      notice.navigate = true;

      // set whether it should be brought to attention
      notice.alert = true;

      // post notice
      noticeID = gShell.addNotice(notice);

    // receive notification when notice is dismissed
      function onNoticeEvent(event, noticeData, appData, appID)
```

Use Notices For

- To notify the user of data changes that they are concerned about

Don’t Use Notices For

- Displaying errors
- Connection notifications
- Be polite with notices
  - Don’t add a notice that’s already been dismissed
  - Reuse existing notices rather than created another

Create Custom Preferences For

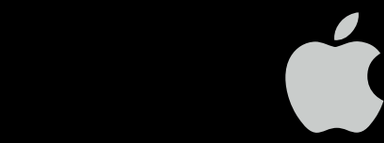
- Users to configure their Notices
- Users to configure Tile usage
- Controlling what the Agent is doing
- Per app settings that do not apply globally

Don’t Use Preferences For

- When preferences already exist in Central prefs

Offline Data Management

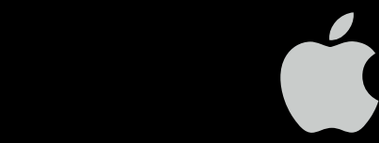
- Shared Object storage
  - Size limit is per domain (default = 100k)
  - Never expires unless user alters settings
  - Best for internal, already parsed data
  - Stored as text
- Local Internet URL storage
  - One size limit for all of Central (default = 20meg)
  - Expiration date (coming in Beta 2)
  - Can store any URL request (SWFs, JPGs, etc.) in its natural format
  - Can store server side query strings as well



addToLocalInternetCache() API

- Download begins once addToLocalInternetCache(url) is called
- Access URL through normal ActionScript methods
- Any attempt to access that URL goes to cache, even if newer data exists on server

- In Beta 1, each request is a new HTTP stream
- In Beta 2, requests are queued and handled serially
- Make responsible use of application's bandwidth

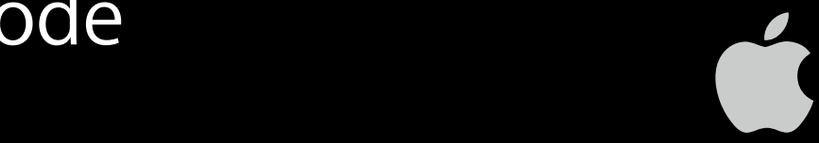


Show Data Loading Progress

- Without visual feedback that something is happening, app can appear slow or hung
- Shell.setProgress(percent) increments progress bar on bottom of shell (pass—1 to show indeterminate progress)
- Shell.setStatus(message) sets text message next to progress bar
- Use loading animation as necessary (such as Tiles use when loading)

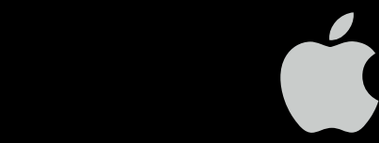
Offline Behavior

- Tell users what's available when offline
  - onNetworkChange(online) callback indicates user has selected offline mode
  - Visually show user what is not available in this mode using mDataGrid.setEnabledIndices(itemArray, state)
  - Disable unavailable functionality using mPushButton, setEnabled(enable), or use an alternate UI screen
- Fail gracefully when data requests fail
  - The user must select the offline mode; Users will forget to do this when offline and attempt to use your app anyway
  - If you app fails, perhaps suggest to user that they check the connection and/or select offline mode



Sharing Data

- Between an App, its Tile(s), and its agent
  - Use LocalConnection, or access the same (cached) URLs
  - Don’t use shared objects to communicate; File access rules prevent it from being open by two SWFs at the same time
  - Centralize SO data access in one place. If Tile and App both need access to the same SO data, put all data management in an agent
- Between different applications and tiles
  - If designed to work together, you can use LocalConnect to that app’s agent



How to Update Your Application

- Update the product version number in the product.xml file on your site
- Central checks for updates to your application once a week

Demo

Michael Williams
mwilliams@macromedia.com

Wrap Up

State of the Project

- Currently in Beta
- Undergoing usability and interface audit
- Public release of SDK—summer, 2003
- Launch of Central platform for end-users—late summer, 2003

Visit the Macromedia Central Developer Center
www.macromedia.com/devnet/central/

To learn how to sign up for
Beta SDK and for regular updates

Who to Contact

Dave Calaprice
dave@macromedia.com

Michael Williams
mwilliams@macromedia.com

Alex Wright
alex@agwright.com

Guillermo Ortiz
QuickTime Technology Manager
quicktimeman@apple.com

http://developer.apple.com/wwdc2003/urls.html

For More Information

Documentation

Macromedia Central Web Site

http://www.macromedia.com/software/central/

Central Developer Center

http://www.macromedia.com/devnet/central/

Beta SDK Sign-up

http://www.macromedia.com/bin/sdk_beta.cgi

For More Information

Documentation

General Information

http://www.apple.com/quicktime

Developer Information

http://developer.apple.com/documentation

QuickTime Information

http://developer.apple.com/QuickTime

Do Not Miss!

Lab: QuickTime Content Creation

Meet with QuickTime content experts for daily content specific topics that are of special interest to you. Bring your own media and PowerBook. No appointment is needed—first-come, first-served.

Sacramento Room

Lab: QuickTime Development

Meet with Apple's QuickTime engineers. Each day will emphasize a specific QuickTime development topic. Bring your own code and PowerBook. No appointment is needed—first-come, first-served.

Sacramento Room

Lab Hours

Monday	12:00pm to 6:00pm
Tuesday	9:00am to 6:00pm
Wednesday	9:00am to 6:00pm
Thursday	9:00am to 4:30pm
Friday	9:00am to 6:00pm

Q&A

Ursula Ringham
Partnership Manager
Worldwide Developer Relations

http://developer.apple.com/wwdc2003/urls.html

WWDC 2003

Worldwide Developers Conference